**IN THE CLAIMS**

1. [Previously amended, Currently amended] An automated software production tool for generating a complete, correct, non ambiguous specification in a formal language which defines a complete operable software program, comprising:

a software-generating computer programmed to:

present user interface mechanisms to allow a user to enter input and receiving and storing ~~receive~~ user input data that defines a plurality of primitives which together define a conceptual model which models a complete system for which a complete computer program is to be automatically written and displaying at least some of said primitives of said conceptual model graphically, said primitives of said conceptual model defining an object model which specifies the system class framework which specifies constant, variable and derived attributes, a set of services including private and shared events and local transactions and integrity constraints specified for the class, and derivation expressions corresponding to derived attributes, and a functional model which specifies dynamic formulates related to evaluations where the effect of events on variable attributes is specified, and a dynamic model which specifies a state transition diagram for each class which specifies service preconditions which are formulas labeling state transitions and a process definition of a class which specifies a template of valid object lives and an object interaction diagram which specifies trigger relationships and global transactions;

convert said conceptual model into a formal language

specification which is a high level repository of ~~formal specification~~ statements defining said conceptual model and stated in a formal language having rules of syntax and semantics;

using said computer programmed with a validator program, automatically validating said formal language specification using the rules of syntax and semantics of said formal language, and verifying that every statement in said formal language specification is syntactically complete, semantically correct and not ambiguous, and if one or more errors are found, displaying information that will indicate which errors were found so as to prompt a user to change said primitives stored as part of said conceptual model to correct said one or more errors so as to generate a validated formal language specification.

2. [currently amended] The system according to claim 1, wherein said software-generating computer is programmed to receive said user input using a CASE tool program for presenting a graphical user interface to allow a user to input the requirements of said formal language specification using a graphical user interface tools to create ~~model~~ said conceptual model graphically.

3. [currently amended, previously amended] The system according to claim 1 wherein said software-generating computer is further programmed with a system logic translator which controls said computer to process said validated formal language specification into one or more computer programs that can control a computer to carry out system logic functions modeled in said object model, functional model and dynamic

model encoded in formal language statements in said formal language specification.

4. [currently amended] The system according to claim 3 wherein said software-generating computer is further programmed with one or more programs which control said computer to present user interface mechanisms which a user can invoke to define instances of primitives of a presentation model which is part of said conceptual model, said primitives then being converted into part of said formal language specification and defining a desired user interface for software which is modeled by said conceptual model and which is to be automatically generated by said software-generating computer, and wherein said software-generating computer is further programmed with a user interface translator which controls said computer to process said portions of said formal language specification defining said desired user interface into one or more computer programs that can control a computer to implement said desired ~~a particular~~ user interface for said one or more computer programs ~~the program~~ created by said system logic translator ~~which has been modeled in said formal specification~~

5. [currently amended] The system according to claim 3 wherein said software-generating computer is further programmed with a database generator which controls said computer to process statements in said formal language specification that define said object model into a data structure or database which is capable of storing the values of at least all attributes of all objects defined in said object model of said formal specification in a manner such that the program or programs generated by said system logic translator can control a computer to read the values of said attributes at any time or store new values for said attributes at any time.

6. [currently amended] The system according to claim 1 wherein said software-generating computer is further programmed by a documentation generator for producing documentation for the software application based on ~~the~~ said formal language specification.

7. [currently amended, previously amended] A process for automatic software generation to create a full, operable desired computer program, comprising:

displaying on a computer user interface mechanisms which can be used by a designer of said desired computer program to create instances of primitives defining a conceptual model of said desired computer program, said conceptual model comprised of an object model, a functional model, a dynamic model and a presentation model, and

receiving user input via user interface mechanisms of said ~~a~~ computer, said user input defining instances of desired primitives of said object model, functional model, dynamic model and said presentation model of said ~~a~~ conceptual model which encodes the requirements ~~a~~ said desired computer program to be automatically generated must meet and the functions ~~the~~ said desired computer program to be generated must perform;

automatically converting said conceptual model into a formal language specification which is a high level repository of formal language ~~specification~~ statements in a computer file which are stated in a formal language having rules of syntax and semantics; and

using said rules of syntax and semantics of whatever formal language in which said formal language specification is written to automatically validate said formal language specification to ensure that said formal language specification is syntactically complete, semantically correct and not ambiguous and if one or more errors are found, to display information to allow a user to adjust the primitives recorded as part of said

conceptual model to correct said error(s) so as to generate a validated formal language specification.

8. [currently amended] The process of claim 7 further comprising the step of using a computer programmed to translate said validated formal language specification into complete, operable, bug-free working system logic computer code that can control a computer to meet the requirements and perform the functions defined in said formal language specification.

9. [currently amended] The process of claim 8 further comprising the step of using a computer programmed to translate said validated formal language specification into computer code that can control a computer to provide a user interface specified in said presentation model of said formal language specification.

10. [currently amended] The process of claim 8 further comprising the step of using a computer programmed with a database translator to translate said validated formal language specification into a data structure or database storing ~~at least~~ the ~~initial~~ values of attributes for objects defined in said formal language specification.

11. [currently amended] The process of claim 8 further comprising the step of using a computer programmed to translate said validated formal language specification ~~specifcation~~ into one or more requested files of documentation of the program or programs generated by said automatic software generating process.

12. [currently amended] The process of claim 7 wherein the step of receiving

user input comprises presenting user interface mechanisms including menu choices and dialog boxes of a graphical user interface so as to present ~~having~~ tools by which a user may enter data defining said primitives of said conceptual model. ~~define the requirements and functions required of the program to be generated using graphical objects displayed on a computer display so as to present~~

13. [currently amended, previously amended] A computer readable medium containing computer-readable instructions for controlling a computer to aid a user in a process for automatic computer program ~~software~~ generation, said computer-readable instructions, when executed by a computer controlling said computer to: ~~to automatically write code for a complete, operable computer program, by:~~

displaying user interface mechanisms including menu choices and dialog boxes which a user can use to enter ~~receiving~~ user input data defining primitives of an object model, a functional model, a dynamic model and a presentation model, all of which combine to define a conceptual model that defines the requirements and functions of a complete, operable computer program to be automatically generated by said computer, and receiving and storing data entered by said user defining the desired primitives that comprise said conceptual model; ~~written~~;

automatically converting said primitives of said conceptual model into a formal language specification comprised of a plurality of statements written in a formal language having predefined rules of syntax and semantics;

using said predefined rules of syntax and semantics to automatically ~~asutomatically~~ validate said formal language specification to ensure that it is syntactically complete, semantically correct and not ambiguous, and if one or more errors are found, displaying information to a user to prompt said user to make corrections in said stored

primitives to correct said conceptual model so as to eliminate said errors and generate a validated formal language specification; and

automatically translating said validated formal specification into one or more computer programs that implement all the system logic, data structures, if any, and user interface functionality, if any, encoded in said formal specification.

14. [currently amended, previously amended] A computer readable medium containing computer-readable instructions for controlling a computer to aid in a process for automatic software generation said computer-readable instructions ~~to automatically write code for a complete, operable computer program, by~~:

displaying user interface mechanisms including menu choices and dialog boxes which a user can use to enter ~~receiving~~ user input data defining primitives of an object model, a functional model, a dynamic model and a presentation model, all of which combine to define a conceptual model that defines the requirements and functions of a complete, operable computer program to be automatically generated by said computer, and receiving and storing data entered by said user defining the desired primitives that comprise said conceptual model; ~~written~~;;

automatically converting said conceptual model into a formal language specification comprised of statements in a computer file written in a formal language having predefined rules of syntax and semantics;

using said predefined rules of syntax and semantics to automatically validate said formal specification to ensure that it is syntactically complete, semantically correct and not ambiguous, and if one or more errors are found, displaying information to a user to prompt said user to make corrections in said stored primitives to correct said conceptual model so as to eliminate said errors and generate a validated formal language

specification.

15. [currently amended, previously amended] A method for automatically generating a desired computer program ~~software application~~, comprising:

displaying on a computer user interface mechanisms which can be used by a designer of a desired computer program to create instances of primitives defining a conceptual model of said desired computer program, said conceptual model comprised of an object model, a functional model, a dynamic model and a presentation model, and receiving user input via said user interface mechanisms that define one or more instantiations of said primitives that define said object model, said functional model, said dynamic model and said presentation model, which together fully define the functionality and interface of said desired computer program, and storing said instances of said primitives; ~~presenting a graphical user interface to allow a user to input requirements for a software application to be automatically generated including user interface patterns;~~

automatically converting said primitives of said object model, said functional model, said dynamic model and said presentation model into statements of ~~generating~~ a formal language specification written in a formal language having predefined rules of syntax and semantics and storing said formal language specification statements ~~for said software application based on said inputs received from said user~~;

using said rules of syntax and semantics, automatically validating said formal language specification to ensure ~~insure~~ it is syntactically and semantically correct and complete and not ambiguous and if one or more errors are found in said formal language specification, displaying information to allow a user to adjust the primitives recorded as part of said conceptual model to correct said error(s) so as to generate a validated formal language specification; and

automatically generating said desired computer program ~~software application~~ based on statements in said formal language stored as said ~~the~~ validated formal language specification using one or more translator programs that automatically convert statements in said formal language which are part of said ~~translate the~~ validated formal language specification into computer ~~working~~ code in a target computer programming language.

16. [currently previously amended] The method according to claim 15, wherein:

said formal language specification includes statements defining an application user interface to be presented to a user ~~of the software~~ of the software application; and

the step of generating said desired computer program ~~the software application~~ includes automatic generation of computer programming language instructions from formal language statements generated from said primitives of said presentation model, said instructions for controlling a computer to present a ~~the~~ user interface in accordance with ~~the~~ patterns specified in said portions of said formal language specification derived from said presentation model.

17. [currently amended, previously amended] A process for forming a functional model that mathematically or logically relates events to the values of attributes the events act upon as part of the definition of the desired functionality of a computer program to be automatically generated using, among other things, said functional model, comprising the steps:

presenting a dialog box or any other form of user interface that allows a user to specify a class, and an attribute of that class and an event of that class and

allows the user to specify mathematical or logical expressions which define a condition and an ~~action~~ effect, said condition being an expression which, when evaluated, will result in a true or false logical value, and said effect ~~action~~ being a mathematical or logical expression which will generate a new value for the specified attribute in a data type compatible with the data type of said attribute;

receiving user input via said dialog box that specifies instantiations of primitives that define ~~to specify~~ said class, any attributes and events of said class and that specifies ~~to specify~~ a primitive of said functional model in the form of a single mathematical or logical expression which defines said condition and results in a value which can be mapped to one of only two possible values~~:~~ which are true or false, and which specifies ~~to specify~~ a functional model primitive in the form of a single mathematical or logical expression which defines said action and results in a value for said specified attribute in a data type which is compatible with a ~~the~~ data type of said specified attribute.

18. [currently amended, previously amended] A process for forming a functional model that mathematically or logically relates events to the values of variable attributes the events act upon as part of the definition of the desired functionality of a computer program to be automatically generated using, among other things, said functional model, comprising the steps of:

presenting a dialog box or any other form of user interface that allows a user to specify a class, and ~~an~~ a variable attribute of that class and an event of that class and allows the user to specify mathematical or logical expressions which defines an effect ~~action~~, said effect ~~action~~ being a mathematical or logical expression which will generate a new value for the specified attribute in a data type compatible with the data

type of said attribute when said event is executed;

receiving user input via said dialog box or other form of user interface mechanism to specify said class, attribute and event of said class and to specify a single mathematical or logical expression which defines said effect ~~action~~ and results in a value for said specified attribute in a data type which is compatible with the data type of said specified attribute.

19. [currently amended, previously amended] An apparatus for forming an object model and a functional model as part of a conceptual model that defines desired classes of objects and functionality and a user interface of a desired computer program, said functional model functioning to ~~that~~ mathematically or logically relate~~s~~ events to the values of variable attributes the events act upon, comprising:

a computer programmed to do the following functions:

present to a user a dialog box or any other form of user interface that allows a user to specify a class, and an attribute of that class and an event of that class and allows the user to specify mathematical or logical expressions which define a condition and an effect ~~action~~, said condition being an expression which when evaluated will result in a true or false logical value, and said effect ~~action~~ being a mathematical or logical expression which will generate a new value for the specified attribute in a data type compatible with the data type of said attribute; and

receive user input via said dialog box or other form of user interface mechanism to enter data which specify

instantiations of primitives of a desired object model and a desired functional model as part of a conceptual model that defines a desired computer program, said primitives defining one or more classes, and at least one variable attribute and event of at least some of said classes and said primitives also specifying ~~to specify a single~~ one or more mathematical or logical expressions which ~~define~~ one or more of said conditions in at least some of said classes, ~~and~~ each of said mathematical or logical expressions specified by said primitives resulting ~~results~~ in a value which can be mapped to one of only two possible values~~:~~ which are true or false for any class which has a condition defined, and said primitives defining ~~to specify a single~~ one or more mathematical or logical expressions for said at least some of said classes, each of said mathematical or logical expressions ~~which defines~~ defining an effect ~~said action and~~ which, when evaluated, results in a value for said specified attribute in a data type which is compatible with the data type of said specified attribute.

20. [currently amended, previously amended] An apparatus for forming a functional model that mathematically or logically relates events to the values of variable attributes the events act upon, and for converting said functional model to formal language statements and validating said formal language statements, comprising:

a computer programmed to do the following functions:

present to a user a dialog box or any other form of user interface that allows a user to specify a class, and ~~an~~ a variable attribute of that class and an event of that class and allows the user to specify a mathematical or logical expression which will generate a new value for the specified attribute in a data type compatible with the data type of said attribute when said event is executed; and

receive user input via said dialog box or other form of user interface to specify instances of primitives which define said class, attribute and event of said class and to specify a single mathematical and/or logical expression which defines how the value of the specified attribute will be changed when the specified event occurs ~~executes.~~ ;

automatically convert said primitives into formal language statements in a formal language which form part of a formal language specification which defines a functional model, said functional model defining the functionality of a desired computer program, said formal language being mathematically based and having predefined rules of syntax and semantics;

using said rules of syntax and semantics to validate said formal language statements to ensure that they are complete, correct and not ambiguous, and if any errors are found, for displaying information to a user which will prompt said user to edit said primitives so as to correct said errors.

21. [currently amended] A computer programmed to display ~~a dialog box~~ one or more dialog boxes, menu choices or any other user interface means for entering ~~by which a user can~~ data which defines instances of a plurality of primitives which specify a functional model, an object model, a dynamic model and a presentation model all of which define a conceptual model of at least a portion of a desired computer program, said one or more dialog boxes, menu choices or any other user interface means allowing a user to enter data which define a class, an attribute of that class and a event of that class and ~~to~~ which specify a mathematical and/or logical formula which defines how the value of said specified attribute will be changed when said event occurs, and further programmed to receive user input defining instances of said primitives and automatically convert said instances into statements in a formal language which is mathmatically based so as to have precise, predefined rules of syntax and semantics that make every statement in said formal language capable of being validated to make sure it is complete, correct and not ambiguous and for automatically validating all said statements in said formal language to determine if said statements define a complete, correct and not ambiguous formal language specification of at least part of a desired computer program, said part of said computer program defined by said specification including specification of portions of said desired computer program which implement said object model, said functional model and said dynamic model, and if any errors are found, said computer being programmed to display information which prompts a user to edit said primitives until said error(s) is/are removed and a validated formal language specification exists. ~~s executed~~.

22. [currently amended] The computer of claim 21 wherein said computer is

further programmed to:

automatically translate said validated formal language specification into working computer code in a computer programming language.

~~receive user inputs that define the selections of class, attribute and event and said mathematical or logical formula and encode said user inputs into statements in a formal language specification data structure defining the requirements for a computer program to solve a problem to be solved by a computer program.~~

23. [currently amended, previously amended] A computer readable medium containing a data structure for storing data that defines the valuation formulas in a functional model which is part of a conceptual model which is used by a computer to automatically generate computer code, said functional model identifying the mathematical or logical relationship between events of classes in said conceptual model and the values of variable attributes said events alter when said events occur, said data structure comprising any format data which identifies each event which affects the value of a variable attribute and identifies the attribute affected and defines the mathematical and/or logical operations to be performed when said event is executed or occurs and any operands needed for said mathematical or logical operations.

24. [currently amended, previously amended] An apparatus for creating a graphical user interface to allow user requirements for a computer program to be written by an automated software production tool to be entered and converted to a formal language specification, comprising:

a software-generating computer programmed to:

display a plurality of dialog boxes and/or graphic screens each of which has dialog boxes which can be filled in with data or menu selections, tools or icons which can be invoked to allow a user to enter ~~information~~ primitives defining classes, attributes, services, derivations, integrity constraints, relationships between classes of an object model, and valuation formulas of a functional model for events ~~services~~ that affect the value of variable attributes defined in said object model and a dynamic model and all the other information needed to define a conceptual model of ~~the requirements~~ a computer program to be written by an automated ~~said~~ software production ~~generation~~ tool ~~must comply with~~;

receive user input selections and data that defines said primitives of said conceptual model;

automatically convert said primitives of said conceptual model into formal language statements in a mathematically based formal language which has precise, predetermined rules of syntax and semantics, said formal language statements being stored in a high level repository referred to herein as a ~~of~~ formal language specification; ~~statements stated in a formal language having, predefined rules of syntax and semantics;~~ and

automatically validate said formal language specification using a validator program which, using said predefined rules of syntax and semantics of said formal language, verifies that every statement in said formal language specification is syntactically complete, semantically correct and not ambiguous, and if one or

more errors are detected, for displaying information which prompts a user to edit said primitives so as to correct said errors in said formal language specification so as to generate a validated formal language specification.